

TOSCA / PE2D, RAL (UK), 1986





Demonstration, 1988

Opera 2D update and OperaPy Round-table meeting for magnet designers

Kieran Geiger (TE-MSC-NCM), Ambre Visive (BE-EA-PR) 5th July 2023

Overview

1. Demonstrations (KG)

- 1) Opera 2D 2018 and 2021+
- 2) Python Integrated Design Environment (IDE)
- 3) COMI script conversion
- 2. Presentation: "What could OOP and GIT do for you?" (AV)

3. Strengths and Weaknesses (KG)

4. Group Discussion

- Magnet designer requirements
- Towards an action plan (provisional)



Demonstration: Opera 2D ≤2018

- Models can be generated using GUI or COMI scripting
- Graphical output has integrated COMI editor
- Convenient to create models by recording COMI commands from console





Demonstration: Opera 2D ≥2021

- Models can be generated using GUI, Python script, or legacy COMI script *
- There is a Python console but no editor **
- No longer practical to create models in GUI and then export script ***

SIMULA Opera /d - C./(STStudoSub2012)Example(Opera/20)Sectionsapet/electromagnet.og2_JS	- 🛛 🙀 🗉 electromagnet_static.py - Notepad - 🗆 X
	File Edit Format View Help ""Module for generating the initial E-Core example for the Opera-2d 2020 Examples"" from operacy inport opera2d
P Bodes 09 00 0 Connet 12, 14 (Setur Woodeg Tool Graphing Dat •	Cane 1:::/// Selector: Model Antima def matrix/)+
	Solver Wegnelandes ← □terrort Loos/Nonlose Marchil Ontroe
Regin Tags □ + • Materials □ +	Bendrowy # Set up Magnetostatic analysis type with default settings menom = 2: soon model.analysis_settings.physics_type = opera2d.Physicstype.Magnetostatic
	Adaptive Mesh Cytions # Use CGS units with RZ axisymmetric model model.use cgs units()
Capper • Properties •	Analysis Sanog model.general_settings.symmetry_type = opera2d.SymmetryType.RZ
Matat • Gal 1 • * Bondray CoddL::::: + • Targentaligneet:::: • • * Pad freesarg • * Dorder::: • •	<pre>disc = model.create_rectangla((0.0, 7.0), (10.0, 5.0), name='Disc') former = model.create_rectangla((0.0, 0.0), (5.0, 6.0), name='Coil former') coil = model.create_rectangla((1.0, 0.0), (5.0, 6.0), name='Coil') background_pir = model.create_rectangle((0, -50), (50, 50), name='Background') model.sem_to_back(background_pir)</pre>
PARLAND.1 • • * Raps * Raps * Raps	<pre># Create materials steal_disc model_create_material('Steal') iron_former = model_create_material('Iron') copper = model_create_material('Copper')</pre>
	<pre>steel_disc.permeability = 100 iron_former.permeability = 300</pre>
	<pre># Assign materials disc.regions(0).material = steel_disc former.regions(0).material = iron_former coil.regions(0).material = copper</pre>
	<pre># Volume properties conductor = model.create_properties('Coil 1') conductor.current_density = 22 coil.regione[].properties = conductor</pre>
	<pre># Set mesh size disc.regions[0].mesh_size = 0.5 coll.regions[0].mesh_size = 0.5 former.regions[0].mesh_size = 0.5</pre>
	<pre># Set boundary conditions tan_mag_bc = model.create_boundary_condition('TangentialMagnetic') tan_mag_bc.type = opera2d.BoundaryConditionType.TangentialFieldMagnetic</pre>
	<pre>for edge in background_gir.regions[0].edges: edge.boundary_condition = tan_mag_bc</pre>
	<pre># Set global mesh properties, generate mesh and analyse solution model.general.settings.mesh_size = 10 model.generate_mesh()</pre>
	model.solve('electromagnet.op2_h5', overwrite=True, foreground=True) # Post-oncress solution
	<pre>post = opera2d.get_post_processing_interface(model)</pre>
	<pre>regions_or_piiot alst.regions + tommer.regions, post.calculate_contour_mpifal_expressions #000 ;</pre>
	type=oper=zd.tontourmap;ype.times, name='Flux Lines')
	<pre># Create graph post_polyline = model.create_polyline([(0., 6.5), (10., 6.5)], pp_body=True) graph = opera2d.get_graphing_interface(model) my_buffer = graph.create_buffer_from_fields_on_edges(name='8_on_line',</pre>
	<pre>fields=['82']) my_line = graph.create_line_from_buffer(name='8_line', buffer_name='8_on_line', x_erray_name='R', y_erray_name='82') graph.plot_line(my_line, 'Default')</pre>
	ifname == 'main':
7 Auforde Action: Cricle warrel Dam 🔹 🕄 Cricle to salent promitive (Cricle to leggie anti-scale)	[]-1_13699
📲 🔎 Type here to search 👔 🚛 📑 📰 🛀 🧐 🖏 🐨 🍪 😕 👀 🈘 🍪 🕅	



Demonstration: Python IDE

Integrated Design Environments

- Tool for code development. Includes Python interpreter, source-code editor, console, and debugger.
- Opera supports use of 3rd party IDEs for developing code *
 - No way to integrate IDE and graphical output
- **Programming paradigm** (Chat GPT):
 - COMI (FORTRAN based): FORTRAN follows a procedural approach, focusing on writing step-bystep procedures or subroutines to perform tasks.
 - Python (with OOP): Python's OOP approach organizes code around objects and their interactions, making it easier to reuse and modify code.





Demonstration: COMI Script Conversion

Process

- 1. Run COMI script using Opera 2D 2021+ COMI executor
- 2. Note that much of post-processing functionality does not work
- 3. Keep 'compatible extract' of COMI file
- 4. Recreate non-backwards-compatible COMI commands with OperaPy
 - Taking advantage of Opera 2018 documentation and log files to better understand how COMI commands worked and what inputs/outputs they require/generate
- 5. Create Python script which runs COMI extract and OperaPy commands together



alculat	n ary :e and graph fields or	n a circular arc			
Toolb	utton				
0					
Comn	nand line para	meters			
	Command	CIRCLE			
	Parameter	Default	Function	Function	
	RADIUS	none	Radius of arc.	Radius of arc.	
	P1	none	Azimuthal coo	Azimuthal coordinate at start of arc.	
	P2	none	Azimuthal coo	Azimuthal coordinate at end of arc.	
	XCENTRE	0	X coordinate a	X coordinate at centre of arc.	
	YCENTRE	0	Y coordinate a	Y coordinate at centre of arc.	
	NP	100	Number of ste points.	Number of steps along the arc, i.e. NP+1 points.	
	COMPONENT	POT	Expression of	Expression of field components.	
	TIME	0	For Steady-St at which basic evaluated.	For Steady-State AC solutions only: time at which basic field quantities are evaluated.	
			number	Angle in degrees around ac cycle.	
		1		Amenditude	
			AMPLITUDE	Amplitude.	
			PHASE	Phase angle.	



Ambre Visive Presentation: "What could OOP and GIT do for you?"



Opera 2D Weaknesses and Issues

1. Opera's Python distribution is not robust

- Opera 2023 cannot be linked to 3rd party IDEs (reliant on 2021 version)
- Not possible to link IDE and GUI
- Export Python command (from GUI) is less useful than COMI equivalent

2. Legacy COMI script executor

- Some COMI functions have been lost but this is not well documented
- Does not raise useful errors or output log files
- Very difficult to feed/fetch parameters to/from COMI scripts

3. Opera technical support

- Documentation could be better
- Support portal is good for narrow queries about EM simulation
 - E.g., methods for evaluating forces, formatting field plots, etc.
- Slow responding to tickets about software issued. Unwillingness to acknowledge problems is frustrating



Opera 2D Strengths and Potential

- Established tool at CERN lots of experience and library of models
 - Only useful if there is backwards COMI compatibility!
- Compared to FEMM Opera has some additional features:
 - Non steady state dynamic solver
 - Optimiser module
 - Hysteresis solver
- Python interface
 - Amber has given us a taste of its potential
 - Already popular with engineers and easy to learn



Group Discussion: Magnet Designer Requirements

Wants / needs from Opera 2D?

Immediate needs

- E.g., continued access to some Opera 2018 licenses for COMI debugging
- Brainstorming ideas
 - E.g., repository of converted COMI scripts

Please start by quickly introducing yourself

- Previous experience with Opera 2D, COMI, and Python
- What might you use Opera 2D for?



Group Discussion: Conclusions

• From our discussion are there any clear priorities?

- 1. Immediate concerns
- 2. Medium term (wish list)
- Best way to work with Dassault/Opera?
- Going forwards:
 - Is it useful to set up a quarterly meetings?
 - How could longer term projects be resourced?
- Any other thoughts / business?





home.cern